PROFESSOR: This is [? Doctor ?] MATLAB, Lecture 6, debugging. We will use the function Fibbo2 to demonstrate some basic debugging. The first thing you want to do when you debug is set a breakpoint at an interesting line. We'll set it at the very first interesting line of the function.

Since Fibbo2 is not in the current folder nor in the path, MATLAB pops up this download box and asks us whether we should add it to the path or simply change folder. We're going to change folder. So now that we've changed folder, MATLAB was willing to put a breakpoint. And now if we run Fibbo2 with 3, we will get stopped right here with the green arrow on the breakpoint.

Notice that at this point, n is defined to be 3 because that was the input we get gave Fibbo. F is still not defined, and v is not defined either. After we do this one line, F will be defined. How do we do this one line?

We have to either click on Step, or Step In, or Step Out, or Continue. All these things are slightly different. I'm going to use Step, in this case. So Step means to do this line and step to the next line.

So now it did that and F has been initialized. Let's see-- yes, F is 1 1. Here is the helper. The helper is the function that actually does most of the calculation. If we step over this one, as well, we're almost done.

Now F is already the full 3 vector, and we will accept the last term, Fn, and put it into v. Let's Step, and now we're about to end. Before we end, we can check that v is 2. So 2 is the value that's going to be returned in the next step.

We can Step again, and now this arrow out means that we're about to leave the function when doing nothing else. And if we take one more Step, we're out. The answer is 2, and we're not in the debugging mode anymore.

We never seem to have entered helper. And the reason for that was that we stepped over this line rather than stepping into it. Let's do that again. I'll move the breakpoint and put it here. And now I'll run it.

I can't run the function by pressing this green arrow because I'm not giving it any input when I just run it. If I run it, I'll get an error. Now n is not defined. And if I step over right now, here's my error-- not enough input arguments because n was not defined.

OK, so let's not run it this way. Let's run it this way, but this time I'll give it 5 as an input. OK, so I put my breakpoint here. I'm stopped right here. And now I'm going to not step over-- Step means step over. I'll Step Into. So Step Into means to step into this function call.

So now we're inside the function helper. Notice that this green arrow tells us where we came from-- sorry, this non-green arrow tells us where we came from. And this green arrow tells us where we currently are.

We can see all of these in the stack. Here's the base. This is where we called Fibbo2 here. Here's Fibbo2-- that's what this non-green arrow is. And here's Fibbo2/helper, that's this location right here.

We can switch to whatever we want. So we can switch to the base. Now we're not here and not here. We're actually on the base. And we can see who the variables are here. There are not that many variables. Specifically, n is not a variable, and v is not a variable, and F is not a variable when we're out here. Let's go back to Fibbo2/helper.

So this F, of course, is not going to be true. Step Over this and now it's going to kill helper again. Let's go once in, just to see how this looks. So if we go in, now the stack is becoming more and more complicated.

There are two entries of Fibbo2/helper because helper called helper again. n at this point is 4. So in fact, it's probably going to call it one more time because the number of elements in F is still 2. We have to wait until we reach down to 2.

So even if we Step In again here, now we're in helper three times. Take a look-- where's my stack? At this point, n is still 3. It will happen once more.

And now it's going to all start collapsing back. So now I've called n with 3. Now I'm starting to come back out of it. So F is going to start growing.

So helper has been called, n is 3. So now I'm going to put F3 equals F2 plus F1. Now this is a little bit boring. So I can use Step Out to step out of a function call. So this basically runs until the current element of stack goes out. It will return.

So if I Step Out once, I'll have two functions-- well, four functions left-- twice helper and then

Fibbo2 and base. So Step Out-- let's look at the stack again. Let's Step Out again. Let's look at the stack again-- good.

At this point, F is growing and n is increasing. And each time, I'm filling up the relevant F. So now I'm going to fill up F5-- Step, return. Now I'm done with this function. And I'm exiting the function, so now I'm done here.

Now all the data is done. I can Step. And if you're bored with this, you just want to finish the function, you can just call Continue. And it will run through the rest of the lines unless it hits a breakpoint.

Now let's do this again. While I'm here-- now I'm here again. I'm in the command line. I'm inside this function. I can evaluate all kinds of stuff. I can look at F. I can look at F1 plus F2. I can evaluate all kinds of things and even change things. I can do silly stuff like F1 equals 3. And now F will be starting with 3 and 1.

In the Fibonacci sequence, the results will be a very odd one, indeed. Let's see how this works. I'm going to put a breakpoint here, press continue-- oh, because I ran it. Let me call Fibbo5, now change this to 3, and now press Continue.

OK, so now what is F? F is a Fibonacci sequence but it's not the Fibonacci sequence because it starts with 3 1. And then there a 4, which is the sum of 3 and 1; and then a 5, which is 4 and 1; and then a 9, which is 4 and 5, et cetera. I changed my Fibonacci sequence while it was running. And then that changed, of course, the rest of the operation.

There's another type of breakpoint that you can do that's called a conditional breakpoint. So let's do this one. You notice that we had to wait until we knew that n was 2. We had to go Step In, Step In again and again. This can get boring.

So what we can do, if you right click, you can set a conditional breakpoint. So I want to stop it on this line only if, let's say n equals 3. So now let me exit debug mode. I'm going to call Fibbo with 15, for example.

First it stops here. I'll tell it to Continue. And now it stops here. Now notice this is when n equals 3. We started with n equals 15. So that means that our stack should be huge. Indeed, our stack is huge.

We have Fibbo2 calling the helper, which calls the helper, and the helper, and the helper-- all these call the helper again, and again, and again until n goes down to 3. Now it's going to start popping back up again. All this time F hasn't been changed because this line has never been reached. But now it has reached for the first time.

And if I click Continue, it won't stop at this breakpoint again. It only stops there once when n is 3. And now it ends. Now it knows we're ready, that v is whatever it is-- 610.

And it's done. One more Step-- it's almost done. One more Step, Continue-- and it's done. 610 is the answer. And this is how you can use the debugger.